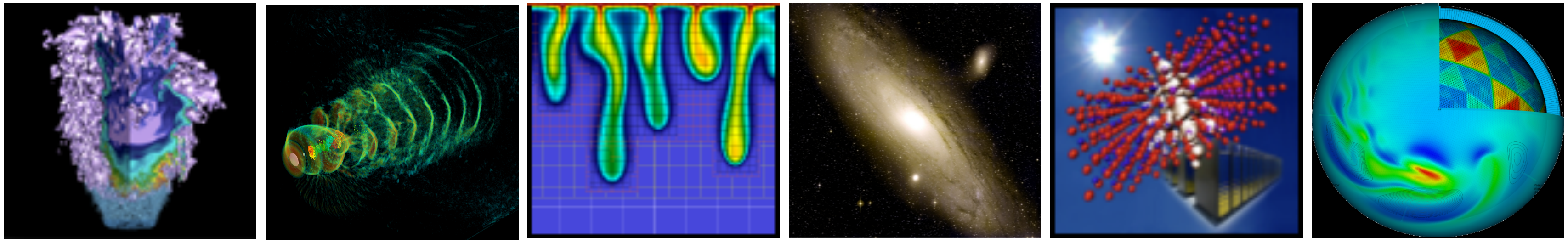


Introduction to Materials Science and Chemistry Applications at NERSC

Zhengji Zhao

NERSC User Services Group

April 5, 2011

U.S. DEPARTMENT OF ENERGY | Office of Science

NERSC National Energy Research Scientific Computing Center

BERKELEY LAB Lawrence Berkeley National Laboratory

Outline

- **Getting started with precompiled materials science and chemistry applications available at NERSC**
- **Out of memory error and parallel scaling**
  - G09
  - VASP
- **Exploring performance benefits from hybrid (MPI/OpenMP) execution on Hopper**
  - Quantum Espresso
- **Summary**

Getting started with the precompiled materials science and chemistry applications available at NERSC

Computing resources at NERSC

NERSC Computational Systems										
System Name	System Type	CPU		Computational Pool					Node Interconnect	Scratch Disk
		Type	Speed	Nodes	SMP Size	Total Cores	Aggregate Memory	Avg. Memory/core		
Hopper II	Cray XE6	Opteron	2.1 GHz	6392	24	153,408	216.8 TB	1.33 GB	Gemini	2 PB
Hopper	Cray XT5	Opteron	2.4 GHz	664	8	5,312	10.6 TB	2 GB	SeaStar	2 PB
Franklin	Cray XT4	Opteron	2.3 GHz	9,572	4	38,288	78 TB	2 GB	SeaStar	436 TB
Carver	IBM iDataPlex	Intel Nehalem	2.67 GHz	400	8	3200	9.6 TB	3 GB	QDR InfiniBand	785 TB
PDSF*	Linux Cluster	AMD/Intel	2+ GHz	~230	2.4	~1000	2.2 TB	2 GB	Ethernet	450 TB
Euclid	Sun Sunfire	Opteron	2.6 GHz	1	48	48	512 GB	10.67 GB	QDR InfiniBand	785 TB

*PDSF is a special-use system hosted by NERSC for the High Energy Physics and Nuclear Science community.

Hopper compute nodes have 24 cores per node

- 6008 32 GB memory nodes, 1.33GB per core
- 384 large memory nodes, 64GB per node, 2.67GB per core

Carver compute nodes have 8 cores per node

- 320 24GB memory nodes, 3Gb per core
- Carver has 80 large memory nodes, 48 GB memory per node, 6GB per core
- Memory limit: soft 2.5GB and 5.5GB; hard 20Gb and 44Gb, respectively

Precompiled materials science and chemistry codes at NERSC

Codes	Hopper	Franklin	Carver
ABINIT	✔	✔	✔
CP2K	✔	✔	✔
CPMD		✔	
Quantum Espresso	✔	✔	✔
LAMMPS	✔	✔	✔
Qbox		✔	
SIESTA	✔	✔	✔
VASP	✔	✔	✔
WIEN2k			✔

Codes	Hopper	Franklin	Carver
AMBER		✔	✔
G09			✔
GAMESS	✔	✔	✔
GROMACS	✔	✔	✔
MOLPRO	✔	✔	✔
NAMD	✔	✔	✔
NWChem	✔	✔	✔
Q-Chem		✔	✔

How to access

- **Modules**
  - An approach that manage user environment for different versions of software
  - Simply use “load” and “unload” to control user environment
  - Commonly used module commands (man module):
    - Module avail - to see available modules
    - Module load, module unload
    - Module list - to see loaded modules list
    - Module show- show what envs defined in the module
- **Modules just define some environment variables in your shell environment if loaded**

How to access

- **Access restrictions:**
  - G09 – just need to agree the license statement
  - VASP – available to the users who own VASP licenses by themselves
- **Some module commands display**
  - module show vasp
    - ls -l /usr/common/usg/vasp/5.2.11/bin
  - module load vasp
    - Which vasp
  - module show g09
    - ls -l /usr/common/usg/g09/b11/g09/*.exel
    - ls –l /usr/common/usg/g09/b11/g09/tests/com

How to run on Carver

Running interactively

Qsub –I –V –l
nodes=2:ppn=8 –q
interactive

```
Cd $PBS_O_WORKDIR
Module load vasp
mpirun –np 16 vasp
```

Running through batch jobs

```
% cat test.pbs
    #PBS -N test_vasp
    #PBS -q regular
    #PBS –l nodes=4:ppn=8
    #PBS –l walltime=12:00:00
    #PBS -j oe
    #PBS -V
    cd $PBS_O_WORKDIR
    module load vasp
    mpirun -n 32 vasp
% qsub test.pbs
```

Note:

Be aware of the parallel job launching scripts in chemistry codes, eg., qchem, molpro, gamess,…, the aprun or mpirun is called inside the launching script.

How to run on Carver

G09 sample job script

```
#!/bin/bash -l
#PBS -N t1
#PBS -q regular
#PBS -l nodes=2:ppn=8,walltime=06:00:00
#PBS -j oe
#PBS -V

mkdir -p $SCRATCH/g09/$PBS_JOBID
cd $SCRATCH/g09/$PBS_JOBID
module load g09
ulimit –Sv unlimited
g09l < $HOME/g_tests/T/t1.inx > $HOME/
g_tests/T/t1.out
ls -l
```

Memory limit on Carver compute nodes:
2.5GB and 20GB for soft and hard memory limit on small memory nodes;
5.5GB and 44GB for large memory nodes

To raise the limit:
For bash/ksh:
ulimit –Sv unlimited
For csh/tcsh:
limit vmemoryuse unlimited

Standard out/error redirection:
avoid file name conflict

Running on scratch file system

- MP2 in g09 can easily fill up your global home quota (40GB)

  Nbasis=694, %Nproclinda=4

```
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-14106.scr-00003
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-17399.scr-00002
-rw-r--r-- 1 zz217 zz217 557263618048 Mar 30 23:56 Gau-10198.scr-00001
-rw-r--r-- 1 zz217 zz217 557272006656 Mar 30 23:57 Gau-9483.rwf
```

- Always run jobs on scratch file system which has much larger quota, 20TB.

Note: Scratch file system is subject to purge, save important results to HPSS archive system.

http://www.nersc.gov/nusers/systems/hpss/

How to run on Hopper

Running interactively

Qsub –I –V –l mppwidth=48 –q interactive

```
Cd $PBS_O_WORKDIR
Module load vasp
aprun –n 48 vasp
```

Running in batch job

% cat test.pbs

```
#PBS -N test_vasp
#PBS -q regular
#PBS –l mppwidth=128
#PBS –l walltime=12:00:00
#PBS -j oe
#PBS -V
cd $PBS_O_WORKDIR
module load vasp
aprun –n 128 vasp
```

% qsub test.pbs

Running on unpacked nodes

Running on 12 cores per node

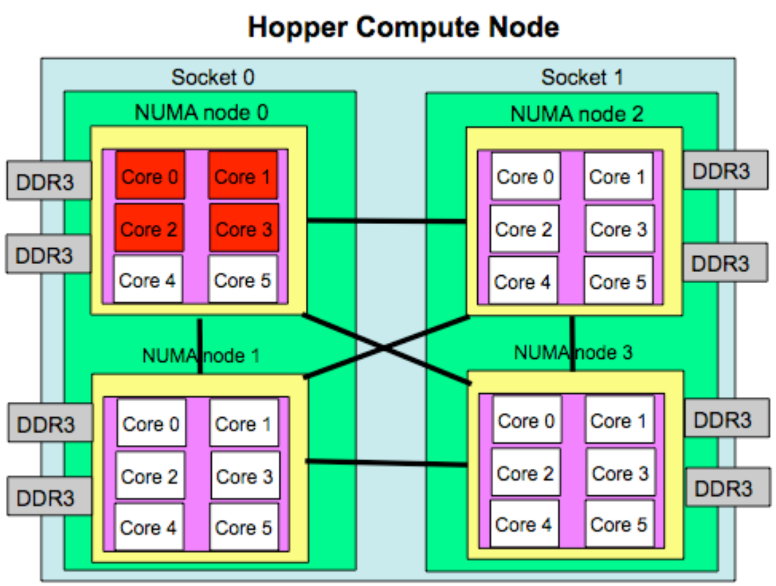
```
% cat test.pbs
    #PBS -N test_vasp
    #PBS -q regular
    #PBS –l mppwidth=768
    #PBS –l walltime=12:00:00
    #PBS -j oe
    #PBS -V

    cd $PBS_O_WORKDIR
    module load vasp
    aprun –n 384 –N12 –S3 vasp
% qsub test.pbs
```

-N12: 12 tasks per node,
-S3: 3 tasks per numa node/socket
man aprun for aprun options

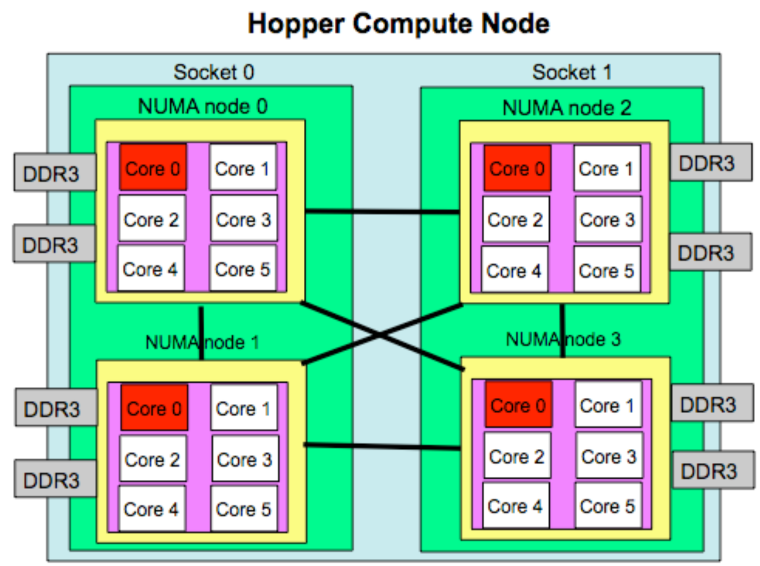
Note: -S option is important when running on fewer than 24 cores on the node. On a vasp job with 660 atom system, ~2.5 times performance difference has been observed.

Optimal MPI tasks placement on a Hopper node

aprun –n128 –N4



aprun –n128 –N4 –S1



Bundle up jobs on hopper

```
#!/bin/bash -l
#PBS -q regular
#PBS -l mppwidth=144
#PBS -l walltime=12:00:00
#PBS -N my_job
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module load vasp
for d in 1 2 3
do
cd dir$d
aprun -n 72 vasp &
cd ../
done
wait
```

This is useful when these jobs have similar run time

Note: A similar job script like this would not work on Carver, because the parallel job launcher mpirun on Carver always starts from the first node allocated regardless if other jobs have used the node or not.

Useful commands

- **qsub, qstat, qalter, qmove**
  - qstat -Qf
  - qalter –l walltime=15:00 jobid
  - qmove debug jobid
- **Showq, checkjob**
  - Showq - approximate priority in the queue
  - Checkjob –v jobid, check job status, can catch some obvious errors

Good practice

- Request the shortest safe wall clock time if possible for a better queue turnaround
- Test job script before submitting a long job
- Check point your jobs if available
- Keep job ids for your jobs

Out of memory error and parallel scaling

Two types of memory errors

- **Memory requirement depends on job types, and implementations**
- **Some codes work within the memory requested (or default memory), G09, NWChem, Molpro, …, etc.**
  - Gracefully exit when memory is not sufficient
- **Others use all the memory available on the node, VASP, Quantum Espresso, LAMMPS, NAMD,…**
  - Killed by the operating system

Parallel scaling issues

- **Running at too high concurrency**
  - Not necessarily reduce the time to solution
  - Code behavior often is not predictable outside of the scaling region
  - Waste resources
- **Running at too low concurrency**
  - Lose productivity unnecessarily
  - Easily run into memory issues

Example 1: G09

- **Request memory in the input file:**
  - Default 32mw=256mb
  - %mem=18gb for SMP+Linda parallel execution
  - %mem=2gb for Linda only parallel execution
- **Parallel execution of G09 has to be requested in the g09 input file**
  - %NprocShared=8
  - %NprocLinda=2
  - If not, jobs run in serial, only 1 core in use, the rest idle
  - Slowdown productivity, wasting computing resource

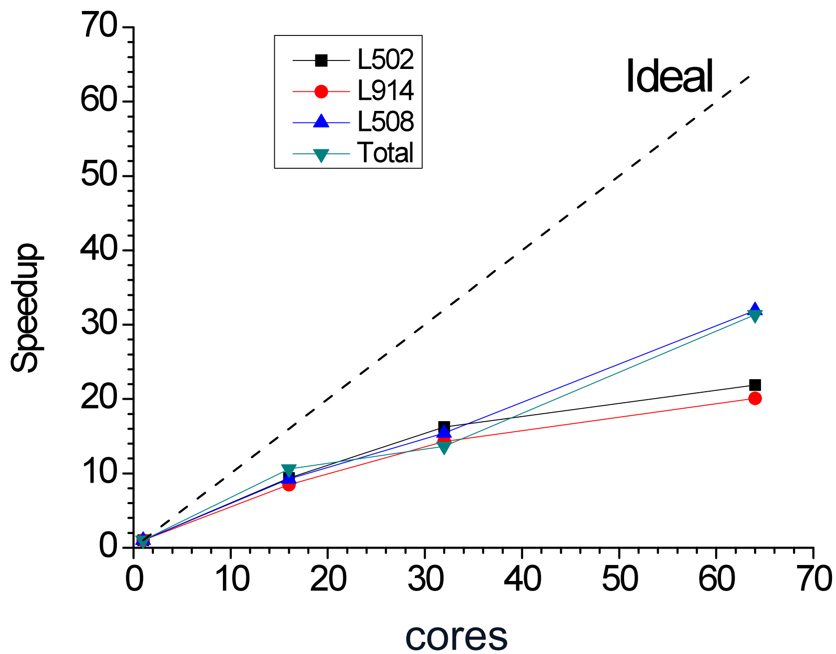
Memory usage of G09

- **G09 provides ways to estimate the memory requirement for various jobs**
  - $M + 2(N_B)^2$ (in 8-byte words), where M is the default 32mw, N_B is the number of basis functions
  - freqmem – determines the memory needed for frequency jobs
- **Link 0 Recommendations:**
  - Run in SMP+Linda parallel
  - %mem=18gb
  - %NprocShared=8
  - %NprocLinda=2
- **G09 reduces threads until fit into memory**

Parallel scaling of G09

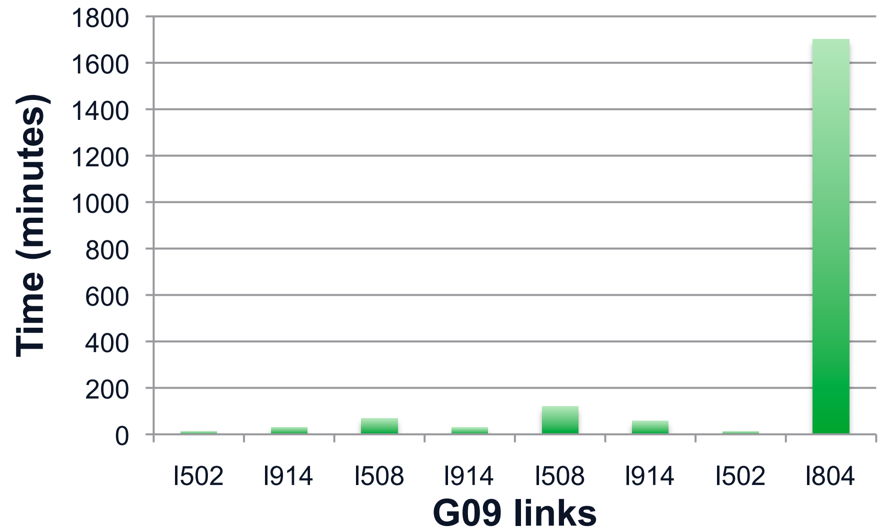
- **G09 runs with a sequence of executables (links), but not all of them can run on multiple nodes**
  - Some of them are Linda parallel (could run on multiple nodes); and some of them are SMP parallel only; and the rest are serial only. 17 out of 79 links are Linda parallelized.
  - Use %Kjob l301 to find out if the sequence of executables (links) that need to run in advance, so to determine if the main component of your calculation can be run on multiple nodes or not.

Parallel scaling of some g09 links



UHF calculation for a system with 61 atoms, NBasis=919

Time spent in each link



UHF calculation for a system with 61 atoms, NBasis=919

Followed by UCIS calculation

NprocLinda=4

Nprocshared=8

Note: Link 804 runs on only one node, other 3 were idling!

* This job hit the wall limit 48 hours while executing l914.

U.S. DEPARTMENT OF ENERGY | Office of Science

Parallel scaling of G09

- When using multiple nodes to run g09 jobs, need to consider if the serial/smp only components are the most time consuming part.
- If yes, then submit separate jobs (dependent jobs) instead of a job with multiple job steps
- Using many nodes to run g09 is not a very good idea, use with caution

Example 2: VASP

- **Memory requirement of VASP**
  - Accurate estimation is difficult
  - NKDIM*NBANDS*NRPLWV*16 –wave function
  - 4*(NGXF/2+1)*NGYF*NGZF*16 – work arrays
  - http://cms.mpi.univie.ac.at/vasp/guide/node90.html
- **If your job run out of memory**
  - Use smaller NPAR if possible. The default NPAR=the number of cores used. But some VASP jobs don't run with reduced NPAR value, eg., hybrid.
  - http://cms.mpi.univie.ac.at/vasp/guide/node139.html

What else we can do

- **Use more cores**
  - so each core needs to store less distributable data
- **Running on reduced number of cores per node**
  - more memory available for each task, especially helpful if the memory to store local data was not sufficient
- **Use larger memory nodes**
  - Trade-off is slow queue turnaround

Running on a reduced number of cores per node on Carver

```
#PBS -q regular
#PBS -l nodes=4:ppn=2
#PBS -l pvmem=10GB
#PBS -l walltime=00:10:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module lod vasp
mpirun -np 8 vasp
```

Process Memory Limits

Type of Node	Soft Limit	Hard Limit
Login Node	2GB	2GB
24GB Compute Node	2.5GB	20GB
48GB Compute Node	5.5GB	44GB

ppn	24GB Node	48GB Node
1	pvmem=20GB	pvmem=44GB
2	pvmem=10GB	pvmem=22GB
4	pvmem=5GB	pvmem=11GB

Running on large memory nodes

Carver

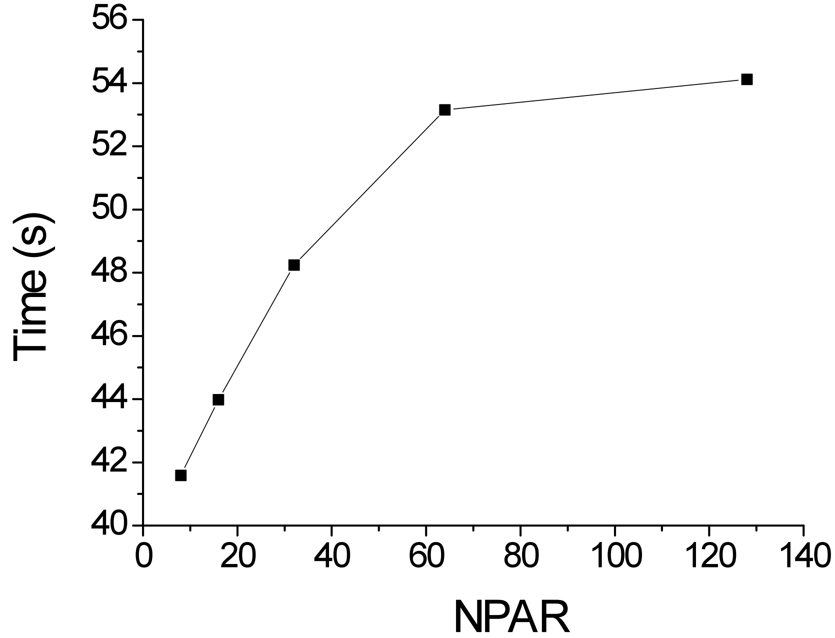
```
#PBS -q regular
#PBS -l nodes=4:ppn=8:bigmem
#PBS -l walltime=00:10:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module lod vasp
mpirun -np 8 vasp
```

Hopper

```
#PBS -q regular
#PBS –l mppwidth=768
#PBS -l mpplabels=bigmem
#PBS –l walltime=00:30:00
#PBS -N test_vasp
#PBS –j oe
#PBS -V

cd $PBS_O_WORKDIR
module load vasp
aprun –n 768 vasp
```

Parallel scaling of VASP

Carver



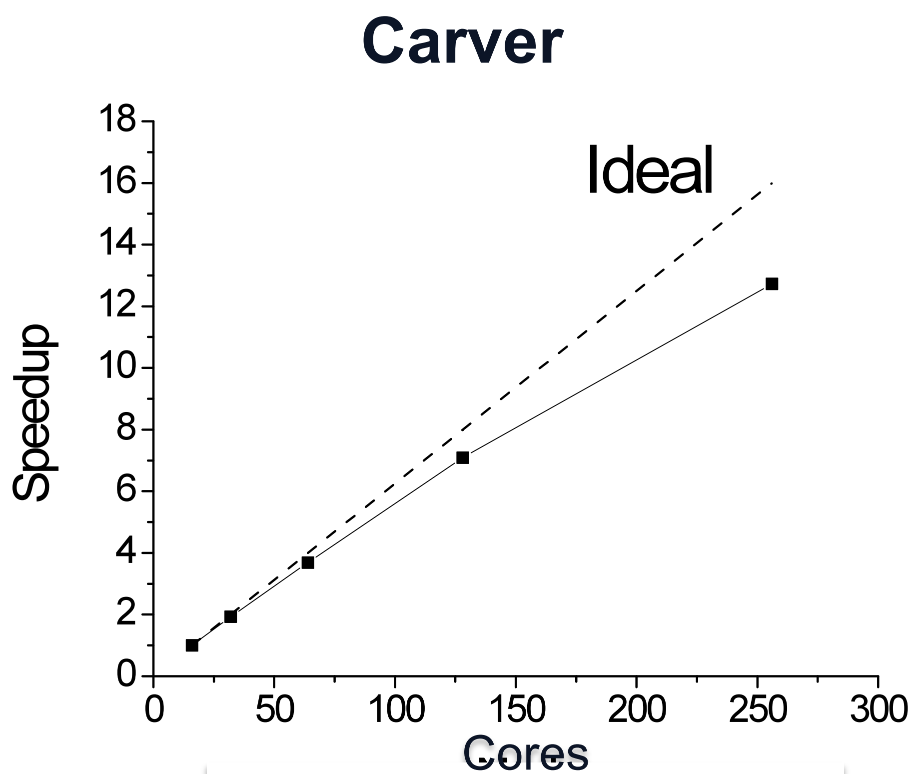
154 atoms,8 k-points, 128 cores used
Time spent in one SC electronic step
*NPAR=2, time=193.13s

Depends on the problem size and job types.

NPAR=8 runs faster for 128 core runs ~ sqrt (number of cores used)

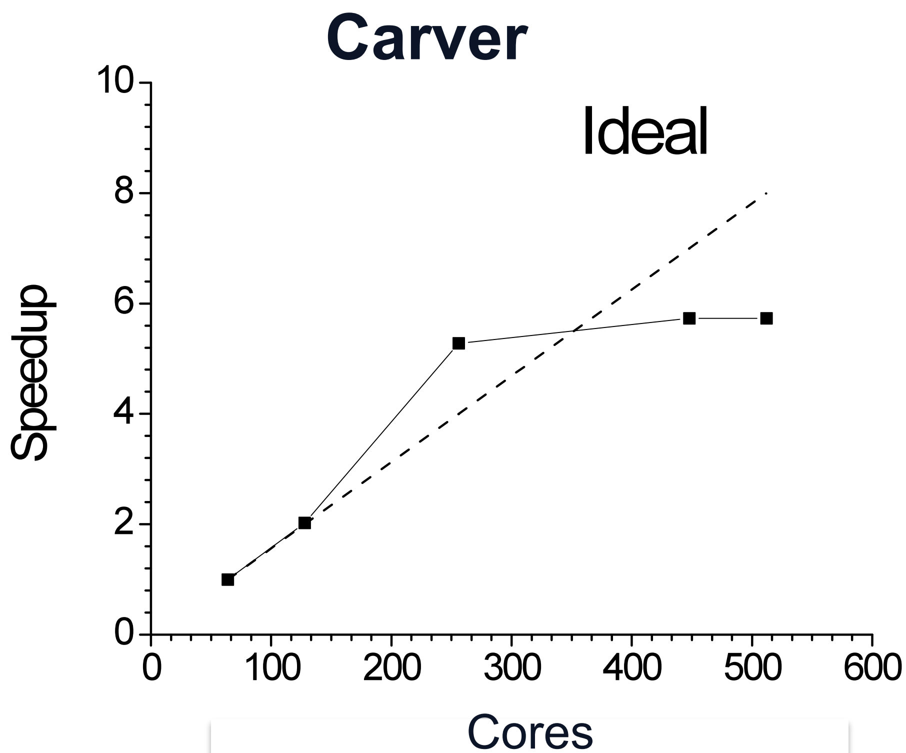
The default NPAR does not run faster than other medium NPAR values

Parallel scaling of VASP



Strong scaling
System with 154 atoms,8-kpoints

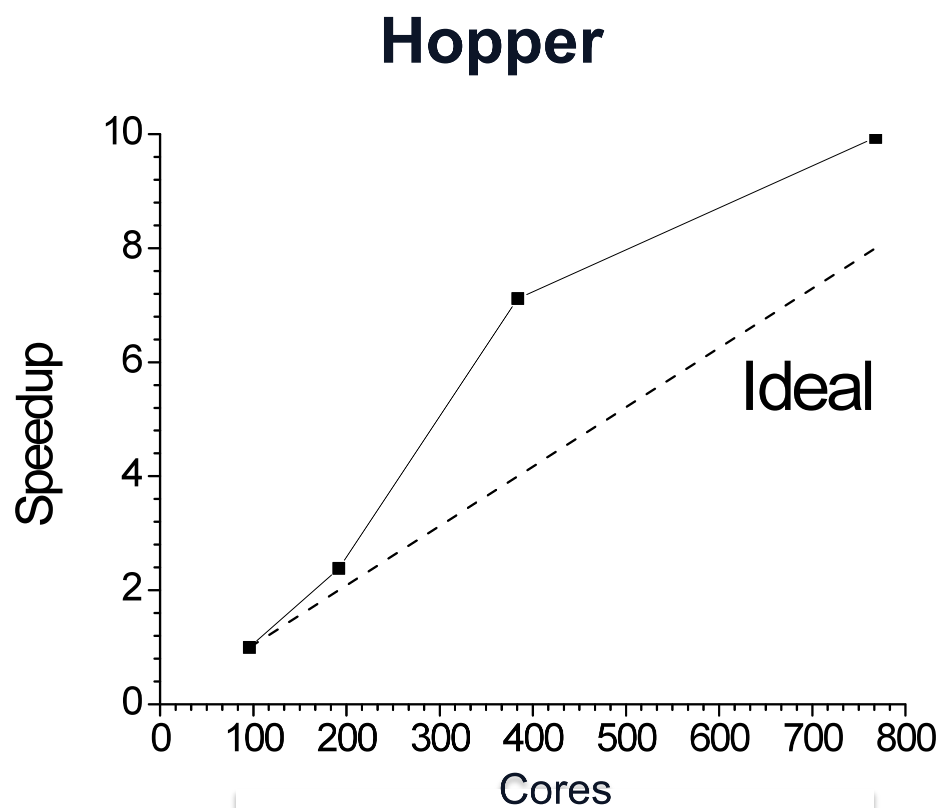
Parallel scaling of VASP



Strong scaling
System with 660 atoms

VASP scales well up to ~500 cores at least for this 660 atom system on Carver

Parallel scaling of VASP



Strong scaling
System with 660 atoms

VASP scales well up to ~800 cores at least for this 660 atom system on Hopper

Parallel scaling of VASP

- VASP (5.2.11) scales well up to near 1 core/atom level both on Hopper and Carver
- When choosing how many cores to use for your job, 1/2 ~ 1 core/per atom would be a good number to start with.
- The scaling of VASP could be affected by many other parameters.

Gamma point only VASP

- **Comparison between gamma-only version and the general k-point version**

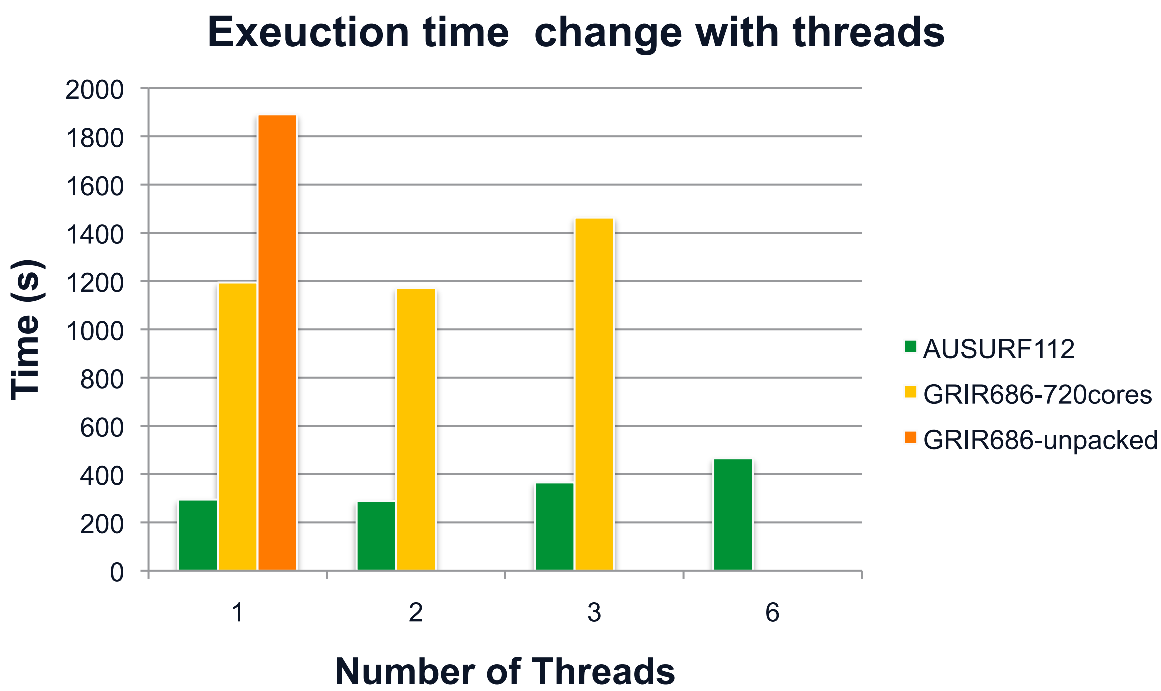
	Memory (GB)	Execution time(s)	WAVECAR size
General kpoint version	0.61	209*	21328479040
Gamma point only version	0.49	204*	10664239520

*Time to execute the second SC step for RMM-DIIS scheme for 660 atom system

- **It is recommended to use the gamma point only version if the system contains only gamma point**

Explore the performance benefits from the hybrid (MPI+OpenMP) execution on Hopper

Hybrid Quantum Espresso on Hopper

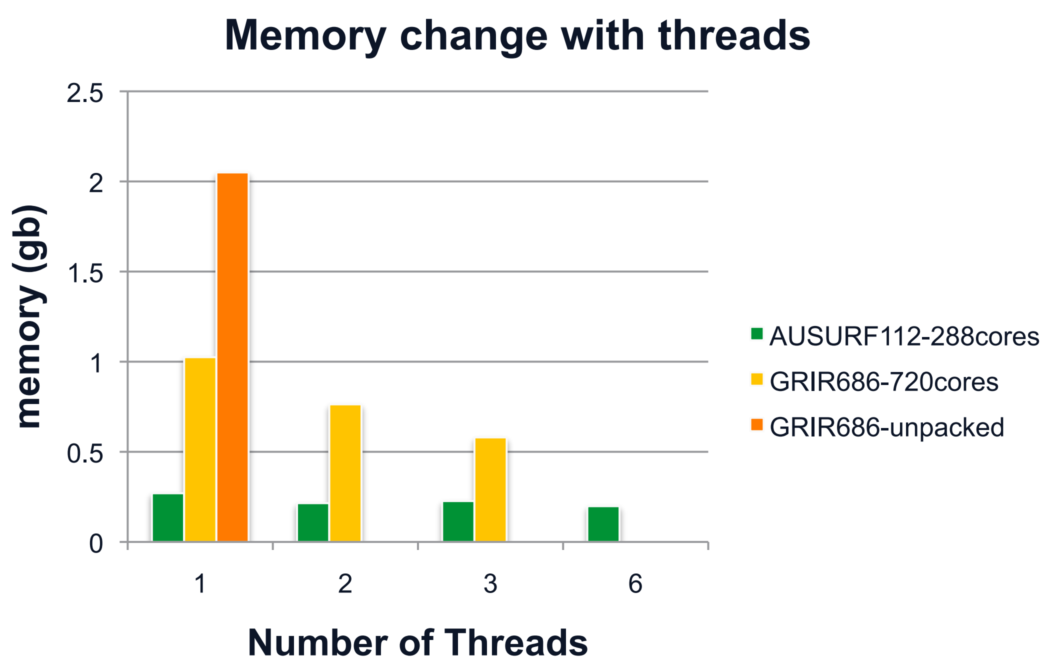


Two benchmark cases were tested with the hybrid (MPI+OpenMP) QE code on Hopper
2 threads per MPI task performs best

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB Lawrence Berkeley National Laboratory

Hybrid Quantum Espresso on Hopper



The memory usage reduces when using more threads

```
#!/bin/bash -l
#PBS -q regular
#PBS -N test
#PBS -l walltime=2:00:00
#PBS -l mppwidth=768
#PBS -j oe
#PBS -V

cd $PBS_O_WORKDIR
module load espresso
export
OMP_NUM_THREADS=2

aprun -n 384 -N12 -S3 -d2 pw.x
-input inputfile
```

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB Lawrence Berkeley National Laboratory

Summary

- **Taking G09 and VASP as examples, addressed two main issues users run into when running jobs at NERSC**
  - For G09 our recommendation is to request the maximum available memory for g09 jobs and not to use a lot of nodes unless you know what you are doing.
  - For VASP jobs that run into out of memory error, in addition to trying NPAR=1 in the VASP input file where applicable, they could be run on more cores and/or on a fewer number of cores per node. Also large memory nodes can be used.

Summary

- **Exploring performance benefits from hybrid execution is recommended on hopper.**
  - hybrid execution reduced memory requirement significantly and could also potentially reduce the time to solution.
- **We didn't address other codes, but the same rule to deal with VASP memory issue will apply to other codes as well.**

- **Recommended readings:**
  - NERSC website, especially
    - http://www.nersc.gov/nusers/systems/carver/running_jobs/index.php
    - https://newweb.nersc.gov/users/computational-systems/hopper/running-jobs/
  - man pages:
    - mpirun
    - aprun
    - qsub, runtime environment variables

- **Ask NERSC consultants questions**
  - Email: consult@nersc.gov
  - Phone: 1-800-666-3772 (or 1-510-486-8600), menu option 3
  - We work with users on a variety of issues
  - Some issues can be solved immediately, others require collaborations for weeks or months